WEST

Help | Logout | Interrupt

Main Menu | Search Form | Posting Counts | Show S Numbers | Edit S Numbers | Preferences

Search Results -

Term	Documents
(13 AND 14).USPT.	5

Database: US Patents Full Text Database
JPO Abstracts Database
EPO Abstracts Database
Derwent World Patents Index
IBM Technical Disclosure Bulletins

Refine Search: Clear

Search History

Today's Date: 12/14/2000

WEST

Generate Collection

L8: Entry 1 of 3 File: USPT Jul 31, 1990

DOCUMENT-IDENTIFIER: US 4945500 A
TITLE: Triangle processor for 3-D graphics display system

DEPR:
Four bits are used to specify a transparency pattern for the triangle. The bits represent a direct encoding of a 2.times.2 transparency screen. This is a repeating mask stippled across the image, with the bit of the mask indexed by the low order x location bit and the low order y location bit. These bits indicate the amount of pixel coverage (which is all or nothing for binary), thus a "1" indicates that this triangle should win; conversely a "0" indicates that this triangle will take a dive. Setting all four bits to one's indicates a non-transparent triangle, zero bits indicate "holes" where triangle from behind (of greater Z) can shine through.

DEPR:
After the bad processors have been mapped out, we are left with what will function as a normal triangle pipe, except with fewer processors. During all subsequent processing the bad processors will stay quiescent, acting as shift registers.

WEST

Generate Collection

L5: Entry 1 of 4 File: USPT Jun 1, 1999

DOCUMENT-IDENTIFIER: US 5909540 A
TITLE: System and method for providing highly available data storage using globally addressable memory

DEPR:
The notions of "node state" and "network state" should be introduced. A node has four states: normal, notified, quiescent, and rebuilding. In the normal state, a node in functioning normally. When notified of another node's failure, the node enters the "notified" state and waits for all its local processing to cease. Once all processing has ceased, the node enters the quiescent state and discards all GRD pages it has cached. When the node receives a "start repopulating" message from the recovery coordinator, it leaves the quiescent state and enters the Rebuilding state. When the node reports to the recovery coordinator that it has completed rebuilding, the node re-enters the normal state.

DEPR:
A fileset deletion operation will deallocate the fileset page, the Inode directory, and the root directory. This transaction begins by marking the fileset as the target of a delete transaction. The fileset page is flushed and the root directory is deleted. Once the root directory is deleted, all free Inodes are deallocated and this step may be repeated as many times as necessary. The super root is then updated to remove the fileset from the super root's set of filesets. Should a node fail at this time, the fileset page will be lost but can be recovered via the GDD callback mechanism. The fileset deletion is completed by deleting the fileset pages. The Inode directory is a set of persistent data structures used to track all the Inodes in a fileset. Using the Inode directory, the file system can locate any Inode in the fileset. The Inode directory contains two major components (1) the free Inode list and (2) the Inode bit map.

DB Name	Query	Hit Count	Set Name
USPT	l13 and l14	5	L15
USPT	thread$ near5 state$1	2801	L14
USPT	l4 and l11	7	L13
USPT	l1 and l11	0	L12
USPT	l6 and l9	42	L11
USPT	l8 and l9	0	L10
USPT	mutual near1 exclusion$1	332	L9
USPT	l3 and l6	3	L8
USPT	l5 and l6	0	L7
USPT	bit near3 mask$1	5700	L6
USPT	l3 and l4	4	L5
USPT	callback	1190	L4
USPT	l1 near5 l2	325	L3
USPT	process$	1443508	L2
USPT	quiescent	19396	L1

WEST

Generate Collection

L5: Entry 2 of 4 File: USPT Mar 10, 1998

DOCUMENT-IDENTIFIER: US 5727209 A
TITLE: Apparatus and method for achieving reduced overhead mutual-exclusion and maintaining coherency in a multiprocessor system utilizing execution history and thread monitoring

ABPL:
A substantially zero overhead mutual-exclusion apparatus and method (90, 120) is provided that allows concurrent reading and updating data while maintaining data coherency. That is, a data reading process executes the same sequence of instructions that would be executed if the data were never updated. Rather than depending exclusively on overhead-imposing locks, this mutual-exclusion mechanism tracks an execution history (138) of a thread (16, 112) to determine safe times for processing a current generation (108, 130, 131) of data updates while a next generation (110, 132, 133) of data updates is concurrently being saved. A thread is any locus of control, such as a processor. A summary of thread activity (106, 122) tracks which threads have passed through a quiescent state after the current generation of updates was started. When the last thread related to the current generation passes through a quiescent state, the summary of thread activity signals a callback processor (104, 124) that it is safe to end the current generation of updates. The callback processor then processes and erases all updates in the current generation. The next generation of updates then becomes the current generation of updates. The callback processor restarts the summary of thread activity and initiates a new next generation of updates. All data-updating threads pass through a quiescent state between the time they attempt to update data and the time the data are actually updated.

BSPR:
This invention is a substantially zero-overhead mutual-exclusion apparatus and method that allow concurrent reading and updating data while maintaining data coherency. That is, a data reading process executes the same sequence of instructions that would be executed if the data were never updated. Rather than depending exclusively on overhead-imposing locks, this mutual-exclusion mechanism tracks a thread (a locus of control such as a processor) execution history to determine safe times for processing a current generation of data updates while a next generation of data updates is concurrently being saved. A summary of thread activity tracks which threads have passed through a quiescent state after the current generation of updates was started. When the last thread related to the current generation passes through a quiescent state, the summary of thread activity signals a callback processor that it is safe to end the current generation of updates. The callback processor then processes and erases all updates in the current generation. The next generation of updates then becomes the current generation of updates. The callback processor restarts the summary of thread activity and initiates a new next generation of updates. All data-updating threads pass through a quiescent state between the time they attempt to update data and the time the data are actually updated.

DEPR:
A CALLBACK 100 is an element of a generation data structure. Each callback 100 tracks a set of elements 102 waiting for safe erasure. Callback 100 may include operational steps specific to a type of elements being tracked.

DEPR:
A CALLBACK PROCESSOR 104 is an entity that monitors a summary of thread activity 106 and processes a current generation 108 of callbacks 100 when it is safe to do so. Then, callback processor 104 causes a next generation 310 of callbacks 100 to become the current generation 308 of callbacks 100 and resets summary of thread

activity 306. A global callback processor may be used to process all callbacks, or multiple callback processors may process individual elements or groups of elements protected by mutual-exclusion mechanism 90.

DEPR:
ERASE means to render the contents of element 102 invalid. Erasure may be accomplished by returning element 102 to a free pool (not shown) for initialization and reallocation, by overwriting, or by moving element 102 to another data structure. The free pool is simply a particular example of another data structure. Any special processing required by a particular element 102 is performed by its associated callback 100.

DEPR:
A QUIESCENT STATE exists for a thread when it is known that the thread will not be accessing data structures protected by this mutual-exclusion mechanism. If multiple callback processors exist, a particular thread can be in a quiescent state with respect to one callback processor and in an active state with respect to another callback processor. Examples of quiescent states include an idle loop in an operating system, a user mode in an operating system, a context switching point in an operating system, a wait for user input in an interactive application, a wait for new messages in a message-passing system, a wait for new transactions in a transaction processing system, a wait for control input in an event-driven real-time control system, the base priority level in a interrupt-driven real-time system, an event-queue processor in a discrete-event simulation system, or an artificially created quiescent state in a system that lacks a quiescent state, such as a real-time polling process controller.

DEPR:
SUMMARY OF THREAD ACTIVITY 106 is a data structure that contains a record of thread execution history that indicates to callback processor 104 when it is safe to process current generation 108 of callbacks 100.

DEPR:
An updater deleting an element removes it from a data structure by unlinking it or using a deletion flag. If current generation 108 is empty, the updater adds a callback 100 containing the element to current generation 108 and causes callback processor 104 to reset summary of thread activity 106. If current generation 108 is not empty, the updater adds a callback 100 containing the element to next generation 110.

DEPR:
An updater changing an element copies it from a data structure into a new element, updates the new element, links the new element into the data structure in place of the original element, uses the callback mechanism to ensure that no threads are currently accessing the element, and erases the original element.

DEPR:
Callback processor 104 interfaces with the quiescence-indicating scheme chosen for summary of thread activity 106 and therefore has various possible implementations. For example, if the quiescence-indicating bits in summary of thread activity 106 have other purposes, no additional overhead need be incurred by callback processor 104 in checking them. Consider a data structure having a dedicated summary of thread activity and a per-thread bit for indicating the occurrence of some unusual condition. Any thread accessing the data structure must execute special-case steps in response to the per-thread bit such as recording its quiescence before accessing the data structure.

DEPR:
Callback processor 104 may be invoked by:

DEPR:
readers just before or just after accessing the data structure protected by mutual-exclusion mechanism 90 (invoking callback processor 104 just after accessing the data structure will incur overhead unless the quiescence-indicating bits in summary of thread activity 106 have multiple purposes);

DEPR:
Callbacks and generations may be processed globally, per-thread where possible, or per some other entity where possible. Global processing is simple to

implement, whereas the other choices provide greater updating efficiency. Implementations that allow threads to be destroyed, such as a processor taken off line, will need to include global callback processing to handle those callbacks waiting for a recently destroyed thread.

DEPR:
A callback processor 124 includes a one-bit-per processor bitmask. Each bit indicates whether its associated processor 16 must be sensed in a quiescent state before the current generation can end. Each bit corresponds to a currently functioning processor and is set at the beginning of each generation. When each processor 16 senses the beginning of a new generation, its associated per-processor context switch counter 122 value is saved. As soon as the current value differs from the saved value, the associated bitmask bit is cleared indicating that the associated processor 16 is ready for the next generation.

DEPR:
Callback processor 124 is preferably invoked by a periodic scheduling interrupt 126. Processor 16 may alternatively clear its bitmask bit if scheduling interrupt 126 is in response to an idle loop, a user-level process execution, or a processor 16 being placed off line. The latter case is necessary to prevent an off line processor from stalling the callback mechanism and causing a deadlock.

DEPR:
When all bits in the bitmask are cleared, callback processor 124 processes all callbacks 128 in a global current generation 130 and all callbacks 128 associated with the current processor in a per-processor current generation 131.

DEPR:
Mutual-exclusion mechanism 120 also includes a global next generation 132 and a per-processor next generation 133. When a particular processor 16 is placed off line, all callbacks 128 in its associated per-processor current generation 131 and per-processor next generation 133 are placed in global next generation 132 to prevent callbacks 128 from being "stranded" while the processor is off line.

DEPR:
Mutual-exclusion mechanism 120 implements callbacks 128 with data structures referred to as rc.sub.-- callback.sub.-- t and rc.sub.-- ctrlblk.sub.-- t and the below-described per-processor variables.

DEPR:
rclockgen: A generation counter that tracks a global generation number. This variable indicates whether a corresponding processor has started processing a current generation and if any previous generation callbacks remain to be processed.

DEPR:
rclocknxtlist: A per-processor list of next generation callbacks comprising per-processor next generation 133.

DEPR:
rclockcurlist: A per-processor list of current generation callbacks comprising per-processor current generation 131.

DEPR:
rclockintrlist: A list of callbacks in the previous generation that are processed by an interrupt routine which facilitates processing them at a lower interrupt level.

DEPR:
A separate copy of data structure rc.sub.-- callback.sub.-- t exists for each callback 128 shown in FIG. 4. The rc.sub.-- callback.sub.-- t data structure is described by the following C-code:

DEPR:
rcc.sub.-- next links together a list of callbacks associated with a given generation;

DEPR:
rcc.sub.-- callback specifies a function 134 to be invoked when the callback

generation ends;

DEPR:
rcc.sub.-- arg1 and rcc.sub.-- arg2 are arguments 136 passed to rcc.sub.-- callback function 134; and

DEPR:
rcc.sub.-- flags contains flags that prevent callbacks from being repeatedly associated with a processor, and that associate the callbacks with a memory pool.

DEPR:
The first argument in function 134 is a callback address. Function 134 is responsible for disposing of the rc.sub.-- callback.sub.-- t data structure.

DEPR:
Mutual-exclusion mechanism 120 implements global current generation 130, per-processor current generation 131, global next generation 132, and per-processor next generation 133 of callbacks 128 with a data structure referred to as rc.sub.-- ctrlblk.sub.-- t that is defined by the following C-code:

DEPR:
rcc.sub.-- mutex is a spin lock that protects the callback data structure (rcc-mutex is not used by readers, and therefore, does not cause additional overhead for readers);

DEPR:
rcc.sub.-- curgen contains a number of callbacks 128 in global current generation 130;

DEPR:
rcc.sub.-- maxgen contains a largest number of callbacks 128 in any next generation 132, 133 of callbacks 128 (when rcc.sub.-- curgen is one greater than rcc.sub.-- maxgen, there are no outstanding callbacks 128);

DEPR:
rcc.sub.-- nxttail is a tail pointer to the rcc.sub.-- nxtlist pointer (note that per-processor lists are used wherever possible, and global lists are used only when a processor having outstanding callbacks is taken off line);

DEPR:
rcc.sub.-- nreg is a counter field that counts the number of "registered" callbacks (a registered callback is one presented to mutual exclusion mechanism 120 for processing);

DEPR:
rcc.sub.-- nchk is a counter field that counts the number of times a rc.sub.-- chk.sub.-- callbacks function has been invoked;

DEPR:
rcc.sub.-- nprc is a counter field that counts the number of callbacks that have been processed;

DEPR:
rcc.sub.-- ntogbl is a counter field that counts the number of times that callbacks have been moved from a per-processor list to a global list;

DEPR:
rcc.sub.-- nprcgbl is a counter field that counts the number of callbacks that have been processed from the global list;

DEPR:
rcc.sub.-- free is a pointer to a list of currently unused callback data structures that cannot be freed because they are allocated in permanent memory.

DEPR:
Function rc.sub.-- callback performs an update 140 using a single rc.sub.-- callback.sub.-- t as its argument. Update 140 adds callbacks to global next generation 132 and per-processor next generation 133 by executing the following

pseudo-code steps:

DEPR:
Callback processor 124 is implemented by a function referred to as rc.sub.-- chk.sub.-- callbacks. Callback processor 124 is invoked by interrupt 126, which is preferably a hardware scheduling clock interrupt referred to as hardclock(), but only if one or more of the following conditions are met: rclocknxtlist is not empty and rclockcurlist is empty (indicating that the current processor is tracking a generation of callbacks and there are callbacks ready to join a next generation); rclockcurlist is not empty and the corresponding generation has completed; or the bit in rcc.sub.-- needctxtmask that is associated with the current processor is set. The latter condition ensures that if there is a current generation of callbacks, the current processor must be in, or have passed through, a quiescent state before the generation can end.

DEPR:
Function rc.sub.-- chk.sub.-- callbacks has a single flag argument that is set if interrupt 126 is received during an idle loop or a user mode, both of which are quiescent states from the viewpoint of a kernel thread. Function rc.sub.-- chk.sub.-- callbacks executes the following pseudo-code steps:

DEPR:
A function referred to as rc.sub.-- intr, invoked in response to the software interrupt, processes all callbacks in rcc.sub.-- intrlist and those in rclockintrlist that are associated with the current processor. Function rc.sub.-- intr executes the following steps:

DEPR:
The function referred to as rc.sub.-- reg.sub.-- gen registers and starts a specified generation of callbacks if there is no active generation and if the currently registered generation has not completed. The rc.sub.-- reg.sub.-- gen function executes the following steps:

DEPR:
In this application, each thread corresponds to a user process, and the quiescent state is a process waiting for user input.

DEPR:
This implementation has a system-wide scope and preferably uses a hierarchical per-thread bitmap, per-level generation counters, a global generation counter, and a thread counter to track the execution histories of a possible large number of processes. The thread counter is decremented when a process exits. If processes can abort, the thread counter must be periodically reset to the current number of threads to prevent indefinite postponement of callback processing.

DEPR:
In this application, the callback processor is invoked by any thread that enters a quiescent state. However, because a user can fail to provide input to a process, a periodic interrupt should also be used to invoke the callback processor.

DEPV:
if a pending callback is already registered, flag an error;

DEPV:
if the pending callback is not registered, flag the callback as registered;

DEPV:
increment the counter rcc.sub.-- nchk; if rclockcurlist has callbacks and rclockgen indicates their generation has completed, append the callbacks in rclockcurlist to any callbacks listed in rclockintrlist;

DEPV:
if rclockintrlist is empty, send a software interrupt to the current processor to process the appended callbacks;

DEPV:
invoke a function referred to as rc.sub.-- cleanup (described below) to invoke the callbacks and release rcc.sub.-- mutex.

DEPV:
if rcc.sub.-- curlist is not empty, move its callbacks to rcc.sub.-- intrtail;

DEPV:
if rcc.sub.-- nxtlist is not empty, move its callbacks to rcc.sub.-- curlist and invoke rc.sub.-- reg.sub.-- gen to indicate that another generation is required;

DEPV:
if rcc.sub.-- intrlist is not empty, send a software interrupt to cause its callbacks to be processed; and

DEPV:
if rclockintrlist associated with the current processor is not empty, move its callbacks to the global rcc.sub.-- intrlist and broadcast a software interrupt to all processors;

DEPV:
if rclockcurlist associated with the current processor is not empty, move its callbacks to the global rcc.sub.-- nxtlist;

DEPV:
if rclocknxtlist associated with the current processor is not empty, move its callbacks to the global rcc.sub.-- nxtlist; and

DEPW:
add the callback to the rclocknxtlist per-processor list;

DEPW:
add the callback to global list rcc.sub.-- nxtlist;

DEPW:
invoke rc.sub.-- reg.sub.-- gen (described below) to register that at least one additional generation must be processed (This step is not performed in the above-described per-processor case, but is performed by rc.sub.-- chk.sub.-- callbacks in response to the next clock interrupt); and

DEPW:
remove the first callback from rclockintrlist and flag the callback as not registered;

DEPW:
invoke the callback; and

DEPW:
remove the first callback from rcc.sub.-- intrlist and flag the callback as not registered;

DEPW:
invoke the callback; and

DETL:
typedef struct rc.sub.-- callback rc.sub.-- callback.sub.-- t; struct rc.sub.-- callback (rc.sub.-- callback.sub.-- t *rcc.sub.-- next; void (*rcc.sub.-- callback) (rc.sub.-- callback.sub.-- t *rc, void *arg1, void *arg2) ; void *rcc.sub.-- arg1; void *rcc.sub.-- arg2; char rcc.sub.-- flags;) ;

DETL:
typedef struct rc.sub.-- ctrlblk (/* Control variables for rclock callback. */ gate.sub.-- t rcc.sub.-- mutex; rc.sub.-- gen.sub.-- t rcc.sub.-- curgen; rc.sub.-- gen.sub.-- t rcc.sub.-- maxgen; engmask.sub.-- t rcc.sub.-- olmsk; engmask.sub.-- t rcc.sub.-- needctxtmask; rc.sub.-- callback.sub.-- t *rcc.sub.-- intrlist; rc.sub.-- callback.sub.-- t **rcc.sub.-- intrtail; rc.sub.-- callback.sub.-- t *rcc.sub.-- curlist; rc.sub.-- callback.sub.-- t **rcc.sub.-- curtail; rc.sub.-- callback.sub.-- t *rcc.sub.-- nxtlist; rc.sub.-- callback.sub.-- t **rcc.sub.-- nxttail; mp.sub.-- ctr.sub.-- t *rcc.sub.-- nreg; mp.sub.-- ctr.sub.-- t

*rcc.sub.-- nchk; mp.sub.-- ctr.sub.-- t *rcc.sub.-- nprc; mp.sub.-- ctr.sub.-- t *rcc.sub.-- ntogbl; mp.sub.-- ctr.sub.-- t *rcc.sub.-- nprcgbl; mp.sub.-- ctr.sub.-- t *rcc.sub.-- nsync; rc.sub.-- callback.sub.-- t *rcc.sub.-- free;) rc.sub.-- ctrlblk.sub.-- t;

CLPR:
3. The method of claim 1 including registering a callback to initiate erasing of the original element.

CLPR:
10. The apparatus of claim 9 including means for registering a callback to initiate erasing of the original element.

L8: Entry 2 of 3 File: USPT Apr 10, 1990

DOCUMENT-IDENTIFIER: US 4916697 A
TITLE: Apparatus for partitioned clock stopping in response to classified processor errors

DEPR:
A second classification of errors is defined as "level two", (L2). This type of error causes only the detecting unit to stop, and to stop as soon as is possible. This error level classification is useful in areas of a processor which, if stopped, do not interfere with or contaminate processes executing in remaining parts of the machine, even if those processes enter a quiescent, or suspended, state. This error level classification is useful in processing system units which conduct serial processes, but which rely heavily on the synchronism provided by a clock signal across module boundaries. Preferably, all modules of the unit are interrupted in synchronism in response to a level two error, which enhances the prospect of recovery, since the state of the unit will closely conform to that existing at the time of error occurrence. In the description following, L1 and L2 errors are not naturally exclusive. Preferably, a unit generating an error can have its operation interrupted on an L2 basis while L1 interruption operations are proceeding.

DEPR:
Each bit in the SCAPTURE register 72 has an associated bit in two of three mask registers, the associated bits indicating into which levels the particular error is to be classified. These registers are labelled L1 MASK 74, L2 MASK 76, and L23 MASK 78. The SCAPTURE register 72, in the preferred embodiment, is 36 bits wide, while the L1 MASK, L2 MASK and L23 MASK registers 74, 76, and 78 are 24, 24, and 12 bits wide, respectively.

DEPR:
The bit pattern stored in the L1 MASK register 74 comprises a conventional digital mask; that is, a "1" in a particular bit position of the mask classifies an error signal in this bit position in the SCAPTURE register 72 as a level 1 error, while a "0" indicates that the error is not to be classified in level 1. In the preferred embodiment, the L1 MASK register 74 applies to the most significant 24 bits of the SCAPTURE register 72. A mask in the L2 MASK register 76 also masks the most significant 24 bit positions in the SCAPTURE register 72 in the same manner as the L1 MASK in the L1 MASK register 74. This permits error indications captured in the SCAPTURE register 72 to initiate both level one and level two clock stopping, when necessary. This might be necessary when, for example, an error correction strategy calls for a serial processor and associated vector processor to stop synchronously and rapidly.

DEPR:
The outputs of the SCAPTURE register 72 and the L1 MASK register 74 are connected to a level one detection circuit (LEVEL 1 DETECT) 80, which compares the registers contents on a bit-by-bit basis to determine whether any error signals in the SCAPTURE register 72 are to be classified as level one errors according to the L1 MASK in the L1 MASK register 74. If a bit in the most significant 24 bit positions of the SCAPTURE register 72 has been set by receipt of an error signal, and that bit is matched by a set bit in the corresponding bit position of the L1 MASK 74, the level one detection circuit 80 will output an L1 signal. Similarly, the outputs of the SCAPTURE register 72, the L2 MASK register 76, and the L23 MASK register 78 are all connected to a level two detection circuit 82 (LEVEL 2 DETECT). Any time an error indication in the SCAPTURE register 72 is masked by a corresponding L2 indication in a corresponding bit position in one of the registers 76 or 78, the circuit 82 outputs an L2 CHECK STOP signal.

DEPR:
As discussed above, SCAPTURE register bits which have their respective mask bits set in the L1 MASK register 74 will raise the unit's L1 signal, which is forwarded, on signal line 35-1 to the machine check collection circuit 52 in the CLKMAINT unit 50 (FIG. 3). As described below, these SCAPTURE bits also cooperate with corresponding inhibiting signals (INH) from an INHIBIT register 83 to inhibit any subsequent level two stop indications from being recognized by the level two detection circuit 82, since it would be possible to stop the unit out of synchronization with other units in the partition if an error indication was classified as a level two level in the cycle following the L1 classification.

DEPR:
The SCAPTURE register 72 allows each error indication to be captured and held, but provides no information as to the sequence of error occurrence. The secondary lockout (SLOCKOUT) register 86, in combination with level lockout gating circuit 84 and a DETECT LOCKOUT circuit 88, performs this function by monitoring the outputs of the SCAPTURE register 72 for the presence of error conditions in much the same fashion as the SCAPTURE register 72 does with error signals. However, instead of locking each bit position as an error indication occurs as in the line lockout circuit, all positions within group levels are blocked in the SLOCKOUT register 86 after detection of the first error in any position. In this regard, the outputs of the SCAPTURE register 72 are fed through the level lockout gating circuit 84 to the SLOCKOUT register 86. In turn, the outputs of the SLOCKOUT register 86 are fed to the detect lockout circuit 88, which also receives the outputs of the L1 MASK register 74, the L2 MASK register 76, and the L23 MASK register 78. When the first clock stop error (classified as L1 or L2) is entered into the SCAPTURE register 72, the error is provided through the level lockout gating 84 to the input of the SLOCKOUT register 86. The error indication bit is compared in the DETECT LOCKOUT circuit 88 with all of the masks in the mask registers 74, 76 and 78. If the error indication occupies a bit position masked by a bit in the L1 or L2 masks, the DETECT LOCKOUT circuit 88 provides a lockout signal to the level lockout gating 84, preventing it from providing any more error signals masked by bits in the L1 or L2 masks. Similarly, the first error indication entered in the SCAPTURE register 72 which is classified as an L3 error enters the SLOCKOUT register 86, and the detect lockout circuit 88 provides an L3 lock to prevent any more L3-classified error indications from entering the SLOCKOUT register 86. Thus, the SLOCKOUT register 86 will capture the first error indication classified as a L1 or L2 error, as well as the first non-clock check stop (L3 error). This permits error detection and recovery procedures to narrow down the investigation into the reason for an error occurrence. Finally, the card error communications collection function is performed by the check reporting circuit 90 which operates in a manner essentially equivalent to the lockout detection circuit 88 to provide an indication of a clock-stopping L1 or L2 error and an L3 error.

DEPR:
Assume now, that error indication bit n rises, signifying an error indication signal being provided by the error indicating unit connected to error input line n. Assume further that L1 MASK bit n is set, as is the nth bit of the SCHKENBL register. In this case, the output of the upper AND gate of AO gate 70 will rise, causing the output of the AO circuit 70 to rise. The level rise will be available on the positive output of cell 72-n of the SCAPTURE register 72. The positive output will be fed back to the AND gate of AO gate 70, preventing any successive error clocked into cell 72-n. In addition, the positive output of the SCAPTURE cell 72-n is fed to a gate combination including AND gate 100 and 24-way OR gate 101. The AND/OR combination 100 and 101 represents the level one detection circuit 80, with the OR gate 101 collecting the outputs of 23 other AND gates in addition to AND gate 100. If the L1 MASK bit corresponding to the nth bit of the SCAPTURE register is set, the AND gate 100 will be activated when the output of SCAPTURE cell 72-n rises. This essentially classifies the nth error indication as a level one error, which is forwarded by way of the OR gate 101 as an L1 signal. Similarly, if the bit in the L2 MASK corresponding to the nth cell of the SCAPTURE register is set, the output of the AND gate 110 will rise, causing the output of the 36-way NOR(N) gate 111 to fall, thereby providing the level two check stop signal in the proper polarity for interrupting the provision of clock signals. The NOR gate 111, the AND gate 110 and 35 other AND gates comprise the level two detect circuit 82 of FIG. 6.

DEPR:
The level lockout gate in circuit 84 is represented by the AO gate 120 whose output is connected to the input of the nth cell of the SLOCKOUT register 86. When an error signal is asserted on the nth error indication line, causing the output of SCAPTURE cell 72-n to rise, the output of the AO gate 120 rises. The rising output is clocked through SLOCKOUT cell 86-n. The output of this cell is fed back through the AO gate 120, which "locks" the cell from responding to any further changes in the output of SCAPTURE cell 72-n. The error indication locked into SLOCKOUT cell 86-n is classified as a level one/level two lockout through the combination of OR gate 122, AND gate 123, and 24-way OR gate 124. Similarly, the AND gate 126 and 12-way OR gate 127 classify the lockout as a level three lockout if the corresponding bit of the L23 MASK is a "0", which, when provided in inverted form to the AND gate 126, appears positive. Lockout detection of the bit in SLOCKOUT cell 86-n is provided by OR gate 130, AO gate 132 and inverter 134.

DEPR:
The INH register 83 of FIGS. 3 and 6 stores an inhibit mask equal in length to the L1 mask, and operating on the same bits in the SCAPTURE register. Whenever one of the inhibit mask bits is set, the clock generator for the affected unit will not respond to an L2 CHECK STOP generated for a corresponding SCAPTURE register bit. For a detailed understanding, refer to FIGS. 4A and 7. Assume inhibit mask bit n is set, indicated by a positive level value for the output of INH register output 83-n (ENABLE L2 INHIBIT n). The output of the AND gate 102 will rise when SCAPTURE bit 72-n is set. This will deactivate the output of NOR gate 103 (INHIBIT L2 STOP), which, since the L2 CHECK STOP and INHIBIT L2 STOP signals are fed to the AND gate 104 in the clock generator (FIG. 4A) the INHIBIT L2 STOP signal will prevent the output (B) of the AO gate 68 from falling. This will limit the clock generator to respond only to the CARD RUN GATE signal for suspending provision of C.sub.1 and C.sub.2.

DEPR:
The CLKMAINT mechanism is shown in detail in FIGS. 8 and 9. In FIG. 8, the outputs of all of the level one detection circuits in the processor units are collected on input lines. Thus, for example, the signal lines 35-1 and 45-1 are shown. According to the L1CHKENBL MASK in the register 150, those L1 signals with a corresponding MASK bit pass through the gating circuit 152 to an L1CAPTURE register 154. The L1CAPTURE register 154 operates in the same manner as the SCAPTURE register described above. That is, it captures any L1 signal arriving on an L1 signal line from a unit, so long as that L1 signal line is enabled by the mask. A lockout loop consisting of a lockout gating circuit 155 and a L1 lockout register 153 captures the first L1 signal entering the L1 capture register 154 on any L1 signal line and then ignores any later arriving L1 signals on that line. The L1CAPTURE register is a t bit register, in which each bit position corresponds to a respective one of a maximum of t SECMAINT circuits.

DEPR:
The output of the L1CAPTURE register 154 is provided to each of four partition detection circuits 156-159. The partition detection circuits also receive at their inputs the output of a primary check stop circuit consisting of SRL pairs 160-0 through 160-t. Each partition detection circuit is for generating a signal identifying one of four predetermined partitions. Each SRL pair is preset with a two-bit code corresponding to one of the four predetermined partitions. Since each SRL provides both a positive and inverted output, each SRL pair of the primary check stop circuit indicates one state of a four-state code. Each primary check stop SRL pair is associated in each partitioned detection circuit with a corresponding bit in the L1 capture register 154. Thus, the output of bit position 0 of the L1 capture register 154 is associated in each of the partition detection circuits 156-159 with primary check stop SRL pair 160-0, and so on. Continuing the example, if a level one error signal is registered into bit position 0 of the L1CAPTURE register 154 from the SECMAINT circuit connected to the input of bit position 0, it will be passed to the input of the partition detection circuit enabled by the code in SRL pair 160-0. The outputs of the partition detection circuits 156-159 provide board partition stop signals, PARTSTOP 0-3. Each PARTSTOP signal indicates that the clocks for each unit of a respective partition are to be interrupted. The partition detection circuit outputs are fed to a stop collection multiplexer 170, which is illustrated in FIG. 9. In FIG. 9, four board partition (BRDPART) registers 172-175 are connected to the inputs of the multiplexer 170. Each of the registers 172-175 has a

partition mask in it in which each bit position corresponds to a respective unit. A mask defines a partition as a group of units whose clocks are to be interrupted in synchronism and in response to an L1 error signal from any member of the partition. The mask comprises a conventional 1-dimensional bit array with 1's entered into the bit position corresponding to partition members, and 0's into every other position. It should be obvious that the bit positions containing 0's are all of the remaining units of the processor which are not included in the partition.

DEPR:
In FIG. 11, the stop collection multiplexer consist of an array of AND gates 220-0 through 220-3, each receiving a respective PARTSTOP signal, as well as bit n from the board partition mask registers 172-175. An OR gate 221 collects the outputs of the AND gates 220-0 through 220-3, and forwards the partition mask bit selected by the AND gate array to a control circuit AND/OR-invert gate 223. The AOI gate 223 controls the input to SRL pair 182-n of the PRUNGATE register 182, and thus controls the state of the nth CARD RUN GATE signal. Also feeding the AOI gate 223 are a START CLKS signal through an inverting AND gate 225 and a STOP CLKS signal through an AND gate 227. The gates 225 and 227 receive the nth bit of the start-stop mask register 210 (FIG. 9). To initialize operation of the apparatus of this invention, the START CLKS signal is raised, causing the START-STOP MASK bit n to be entered into the nth bit of the PRUNGATE register 182. Once initially configured, the CARD RUN GATE signal provided by the PRUNGATE register 182 will continue to be asserted, until changed when an L1 error signal selects a partition mask, which is entered into the PRUNGATE register to determine the state of the CARD RUN GATE signals thereafter.

WEST

(301) 248-0165

Help | Logout | Interrupt

Main Menu | Search Form | Posting Counts | Show S Numbers | Edit S Numbers | Preferences

Search Results -

Term	Documents
(3 AND 4).USPT.	4

Database: US Patents Full-Text Database
JPO Abstracts Database
EPO Abstracts Database
Derwent World Patents Index
IBM Technical Disclosure Bulletins

Refine Search: 13 and 14 Clear

Search History

Today's Date: 12/14/2000

DB Name	Query	Hit Count	Set Name
USPT	l3 and l4	4	L5
USPT	callback	1190	L4
USPT	l1 near5 l2	325	L3
USPT	process$	1443508	L2
USPT	quiescent	19396	L1

WEST

Generate Collection

Search Results - Record(s) 1 through 3 of 3 returned.

☐ 1. Document ID: US 4945500 A

L8: Entry 1 of 3 File: USPT Jul 31, 1990

US-PAT-NO: 4945500
DOCUMENT-IDENTIFIER: US 4945500 A

TITLE: Triangle processor for 3-D graphics display system

DATE-ISSUED: July 31, 1990

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Deering; Michael F.	Mountain View	CA	N/A	N/A

ASSIGNEE INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
Schlumberger Technologies, Inc.	San Jose	CA	N/A	N/A	02

APPL-NO: 7/ 440459
DATE FILED: November 20, 1989

PARENT-CASE:
This is a continuation of application Ser. No. 117,110, filed Nov. 4, 1987, now abandoned.

INT-CL: [5] G09G 1/06
US-CL-ISSUED: 364/522; 340/723
US-CL-CURRENT: 345/422; 345/430, 345/434, 345/506, 345/519
FIELD-OF-SEARCH: 364/521, 364/522, 364/511, 340/721, 340/723, 340/728, 340/729, 340/731, 340/732, 340/747, 340/750, 358/104

REF-CITED:

U.S. PATENT DOCUMENTS

PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
3372557	May 1973	Evans et al.	N/A
3684876	August 1972	Sutherland	N/A
3732557	May 1973	Evans et al.	N/A
3763365	October 1973	Seitz	N/A
3816726	June 1974	Sutherland et al.	N/A
3889107	June 1975	Sutherland	N/A
4283765	August 1981	Rieger	N/A
4291380	September 1981	Rohner	N/A
4343037	August 1982	Bolton	N/A
4371872	February 1983	Rossman	N/A
4380046	April 1983	Fung	N/A
4412296	October 1983	Taylor	N/A
4458330	July 1984	Imsand et al.	N/A
4492956	January 1985	Collmeyer et al.	340/731 X
4550315	October 1985	Bass et al.	340/747 X
4570181	February 1986	Yamamura	N/A
4570233	February 1986	Yan et al.	N/A
4586038	April 1986	Sims et al.	N/A
4646075	February 1987	Andrews et al.	N/A
4658247	April 1987	Gharachorloo	340/747
4697178	September 1987	Heckel	340/723 X
4709231	November 1987	Sakaibara et al.	N/A
4730261	March 1988	Smith	N/A
4736200	April 1988	Ounuma	N/A
4737921	April 1988	Goldwasser et al.	340/723 X

FOREIGN PATENT DOCUMENTS

FOREIGN-PAT-NO	PUBN-DATE	COUNTRY
0137233A2	August 1984	EPX
0168981A2	June 1985	EPX
0167165A2	July 1985	EPX
WO84/01153	July 1984	WOX
WO84/01705	October 1984	WOX

OTHER PUBLICATIONS

"High Speed Image Rasterization Using Scan Line Access Memories", Demetrescu, 1985 Chapel Hill Conference on Very Large Scale Integration 35 (H. Fuchs ed, 1985).
"Super Buffer: A Systolic VSLI Graphics Engine for Real Time Raster Image Generation", Gharachorloo & Pottle, 1985 Chapel Hill Conference on Very Large Scale Integration 35 (H. Fuchs ed, 1985).
"Pixel-Planes: Building a VLSI-Based Graphics System", Foulton et al., 1985, Chapel Hill Conference on Very Large Scale Integration 35 (H. Fuchs ed, 1985).
"Reentrant Polygon Clipping", Sutherland & Hodgman, Communications of the ACM, Jan. 1974, vol. 17.
"An Anaylsis and Algorithm for Polygon Clipping", Liang & Barsky, Research Contributions, Robert Haralick, Editor, 1983 ACM.
"A VLSI Approach to Computer Image Generation", Cohen & Demetrescu, presented at the First Interservice/Industry Training Equipment Conference, Orlando, Fla., Nov. 28, 1979.

ART-UNIT: 217
PRIMARY-EXAMINER: Shoop, Jr.; William M.
ASSISTANT-EXAMINER: Wysocki; A. Jonathan
ATTY-AGENT-FIRM: Carroll; David H. Colwell; Robert C. Haughey; Paul C.

ABSTRACT:

A process which stores a representation of a polygon forming a portion of a three-dimensional object and compares the polygon to pixels from a scan line as they are passed by. The processor stores a representation of a polygon and compares each pixel passed by the processor to the polygon to determine whether the pixel is within the polygon. If the pixel is within the polygon, its Z position (depth) is compared to the Z position of a corresponding position in the polygon. If the Z position of the polygon position is in front of the Z position of the pixel so that the polygon would obscure the previous pixel description, the Z position and an associates material value (e.g., color) of the polygon is substituted for the Z position of the pixel. The three-dimensional object is preferably represented with triangles and each polygon processor is preferably a triangle processor, with a series of triangle processors arranged in a pipeline. Two pipelines may be provided in parallel. Several triangle processors can be placed on a single semiconductor chip with input and output buffers for performing multiplexing, delaying and skewing. An X counter on the chip rederives the pixel X position, eliminating the need for this input.

25 Claims, 16 Drawing figures

☐ 2. Document ID: US 4916697 A

L8: Entry 2 of 3 File: USPT Apr 10, 1990

US-PAT-NO: 4916697
DOCUMENT-IDENTIFIER: US 4916697 A

TITLE: Apparatus for partitioned clock stopping in response to classified processor errors

DATE-ISSUED: April 10, 1990

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Roche; Thomas J.	Endwell	NY	N/A	N/A
Still; Gregory S.	Endwell	NY	N/A	N/A

ASSIGNEE INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
International Business Machines Corporation	Armonk	NY	N/A	N/A	02

APPL-NO: 7/ 211469
DATE FILED: June 24, 1988

INT-CL: [4] G06F 11/00
US-CL-ISSUED: 371/14
US-CL-CURRENT: 714/10; 714/24
FIELD-OF-SEARCH: 371/14, 371/16, 371/15, 371/20, 371/12, 371/68, 371/60, 371/61, 371/7, 371/15.1, 371/16.1, 371/22.6, 371/68.1, 371/68.3, 371/22.1, 364/2MSFile, 364/9MSFile

REF-CITED:

U.S. PATENT DOCUMENTS

PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
3581075	May 1971	Mattsson	371/14

3581075	May 1971	Mattsson	371/14
3593307	July 1971	Gouge	371/14 X
4317200	February 1982	Wakatsuki et al.	371/25
4464751	August 1984	Stranko et al.	371/29
4553204	November 1985	Hashimoto	371/12 X
4587654	May 1986	Matsumoto et al.	371/14
4616335	October 1986	Howe, Jr. et al.	364/900
4679195	July 1987	Dewey	371/29

OTHER PUBLICATIONS

Abstract of Japanese Patent Control System for Error Detection JP 55-32157 and JP 55-32156.

ART-UNIT: 236
PRIMARY-EXAMINER: Ruggiero; Joseph
ASSISTANT-EXAMINER: Beausoliel; Robert W.
ATTY-AGENT-FIRM: Baker, Maxham, Jester & Meador

ABSTRACT:

This apparatus stops the clock for a processor partition consisting of a group of processor units in response to detection and classification of an error in a processor unit which can cause a cascade of errors in the partition group.

14 Claims, 14 Drawing figures

☐ 3. Document ID: US 4625081 A

L8: Entry 3 of 3 File: USPT Nov 25, 1986

US-PAT-NO: 4625081
DOCUMENT-IDENTIFIER: US 4625081 A

TITLE: Automated telephone voice service system

DATE-ISSUED: November 25, 1986

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Lotito; Lawrence A.	Los Angeles	CA	90056	N/A
Huxford; Teresa D.	Los Angeles	CA	90025	N/A
Donaldson; Ann L.	Torrance	CA	90501	N/A

APPL-NO: 6/ 445651
DATE FILED: November 30, 1982

INT-CL: [4] H04M 3/38, H04M 3/50
US-CL-ISSUED: 379/88; 379/196, 379/211
US-CL-CURRENT: 379/88.26; 379/196, 379/211, 379/88.08, 379/88.19, 379/88.24, 902/2, 902/39
FIELD-OF-SEARCH: 179/18B, 179/18D, 179/18DA, 179/5P, 179/6.02, 179/6.17, 179/6.18, 179/6.09, 179/6.11, 360/32, 360/12, 364/513.5, 364/513, 381/36, 381/51, 370/60, 370/61, 370/62

REF-CITED:

U.S. PATENT DOCUMENTS

PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
Re30903	April 1982	Vicari et al.	179/27FH
1922879	August 1933	Burgener	179/27FH
2685614	August 1954	Curtin	179/27FH
2863950	December 1958	Dunning et al.	179/27FH
2892038	June 1959	Gatzert	179/27FH
2985721	May 1961	Gatzert	179/27FH
2998489	August 1961	Riesz	179/6.02
3141931	July 1964	Zarouni	179/6.11
3146310	August 1964	Jeffries et al.	179/6.07
3197566	July 1965	Sanders et al.	179/18BE
3273260	September 1966	Walker	434/307
3296371	January 1967	Fox	381/51
3510598	May 1970	Ballin et al.	179/18BE
3519745	July 1970	Colman	179/5P
3728486	April 1973	Kraus	179/2R
3733440	May 1973	Sipes	179/18B
3920908	November 1975	Kraus	179/2CA
4117270	September 1978	Lesea	179/18BE
4200772	April 1980	Vicari et al.	179/27FH
4210783	July 1980	Vicari et al.	179/18FC
4256928	March 1981	Lesea et al.	179/18BE
4272810	June 1981	Gates et al.	364/900
4302632	November 1981	Vicari et al.	179/27FH
4320256	March 1982	Freeman	179/6.04
4371752	February 1983	Matthews et al.	179/7.1TP

OTHER PUBLICATIONS

"Store & Forward Voice Switching", International Resource Development, Inc., Report #145, pp. 45-56, Jan. 1980.
"A Design Model for a Real-Time Voice Storage System", Hattori et al., IEEE Trans. on Communications, vol. COM-30, No. 1, Jan. 1982, pp. 53-57.
Barish, Bernard T. and Slattery, Paul J., "BISCOM: Rx for Internal Communications", Bell Laboratories Record, vol. 42, No. 6, pp. 175-180 (Jun. 1974).
Watson, Jr., R. E. and S. B. Weinberg, "Telephone Answering Services," Bell Laboratories Record, vol. 43, No. 12, pp. 447-450 (Dec. 1965).
Liske, W., "Remote Controlled Switching of the Telephone Message Service of the Deutsche Bundespost," TN-Nachrichten vol. 70, pp. 13-16 (1970).
Probe Research, Inc., "ECS Telecommunications, Inc., " Proceedings of Voice Processing Seminar, Sep. 15, 1982.
Probe Research, Inc., "Voice Message Service," Proceedings of Voice Processing Seminar, Sep. 15, 1982.
Probe Research, Inc., "Logic Labs, Inc." Proceedings of Voice Processing Seminar, Sep. 15, 1982.
Probe Research, Inc., "BBL Industries, Inc.," Proceedings of Voice Processing Seminar, Sep. 15, 1958.
Probe Research, Inc., "Wang Laboratories," Proceedings of Voice Processing Seminar, Sep. 16, 1982.
Probe Research, Inc., "American Telephone and Telegraph, Inc.," Proceedings of Voice Processing Seminar, Sep. 16, 1982.
Probe Research, Inc., "Commterm, Inc.," Proceedings of Voice Processing Seminar, Sep. 16, 1982.
Probe Research, Inc., "American Express Company," Proceedings of Voice Processing Seminar, Sep. 16, 1982.
Probe Research, Inc. "Equitable Life Assurance," Proceedings of Voice Processing Seminar, Sep. 16, 1982.
Probe Research, Inc., "Massachusetts General Hospital," Proceedings of Voice Processing Seminar, Sep. 16, 1982.

Seaman, John, "Electronic Mail Coming at You," Computer Decisions, pp. 129-160 (Oct. 1982).
"Voice Mail Update," Electronic Mail & Message Systems, vol. 4, No. 20 (Oct. 15, 1980).
Hanson, Bruce L., R. J. Nacon and D. P. Worrall, "Custom Calling Features Cater to Customers," Telephony, pp. 28-32 (Sep. 1980).
"Elect. Mail Pack Unveiled by DEC." Electronic News, vol. 27, No. 1365 (Nov. 21, 1981).
ECS Telecommunications, Inc. Marketing Literature for their UMX System (Jan. 7, 1982).
Memo from C. W. Murphy to Jack Atkin Dated Jan. 30, 1981.
"ECS Unveils 1,000--User Digital Message Exchange," Communications.
Matthews, G. H., "The Pitfalls of Small Telecommunications Trunk Groups," ECS Telecommunications, Inc, (1981).
"New Product, Voice Message Systems," Business Communications Review pp. 37-40 (Jan.-Feb. 1981).
Dukes, A., "IBM Unveils Voice Mailbox; Seen as Step Toward PBX," MIS Week, vol. 2, No. 39 (Sep. 30, 1981).
"Speechfile--IBM's Secret Message System Weapon," Electronic Mail & Message Systems, vol. 5, No. 12 (Jun. 15, 1981).
"Introducing Voice Store & Forward," Computer Decisions, (Oct. 1981).
Out Voice Product Brochures, Voice and Data Systems, Inc.
Dukes, A., "Atlanta Firm Enters Voice-Message Arena," Management Information Systems Week, p. 6 (Nov. 18, 1981).
"New Local Net, Voice Store and Forward from Wang," Computer Decisions (Aug. 1981).
Delphi Delta 1 Telephone Operator's Training Manual (Apr. 1, 1981).
Delphi Delta 1 Voicebank Data Entry Reference Manual (Jul. 20, 1981).
Delphi Delta 1 Voicebank Marketing Literature.
Delphi Delta 1 Specification.
Delphi Delta 1 Standard Processor Module (SPM-1) Specification (Mar. 13, 1978).
Delphi Pascal Programmers Manual (May 22, 1981).

ART-UNIT: 261
PRIMARY-EXAMINER: Brown; Thomas W.
ABSTRACT:

An automated telephone voice service system includes a data store having a plurality of addressable voice storage message baskets defined therein and a control system coupled between the store and a large plurality of telephone lines of a telephone network. An incoming cable may address a particular message basket by entering a code through the telephone keyboard or by a predetermined association with a particular call in line. Upon identification of the message basket the caller is greeted by a client's own voice and invited to leave a voice message which will be recorded in the message basket or given other client information. Upon entry of a personal identification code a caller is granted access to user account functions which include retrieval of voice messages, forwarding of messages to other message baskets or telephone lines, and administrative functions such as the changing of greetings or account operating criteria. Editing commands may be utilized during the recording of voice messages.

74 Claims, 27 Drawing figures

Display Format: FRO Change Format

End of Result Set

L5: Entry 1 of 1 File: USPT Apr 10, 1990

DOCUMENT-IDENTIFIER: US 4916697 A
TITLE: Apparatus for partitioned clock stopping in response to classified processor errors

US PATENT NO. (1):
4916697

Detailed Description Text (20):
Referring now to FIG. 6, the card error check collection circuit 33 and card level detection and reporting circuit 35 in the SECMAINT element 31 of FIG. 3 are shown in more detail. All error signals generated on the card 30 are fed through a line lockout gate circuit 71 to a capture register (SCAPTURE) 72. Each of the signal lines 71-1 through 71-n is connected to one of n module error indicators in the unit being served by the SECMAINT unit. The line lockout gate 71 acts to provide a clear path for each of the error signal lines 71-1 through 71-n into the SCAPTURE register 72. Entry of the first error indication in the SCAPTURE register 72 on any one of the error signal lines 71-1 through 71-n is fed back to the line lockout gate 71 to prevent another error signal being entered into the register on that error signal line. A masking pattern stored in a secondary check enable (SCHKENBL) register 73, is used to mask error indications through the line lockout gate circuit 71. The mask is in a conventional digital format and operates such that a "1" in a respective bit position in the register 73 will enable the line lockout gate 71 to receive an error indication on a corresponding error indication line. A "0" in that mask position will prevent the line lockout gate 71 from responding to an error indication on the corresponding error indication line. Each of the machine error indication lines 71-1 through 71-q is monitored by a respective one of the bit positions of the SCAPTURE register 72, which captures and holds the first active indication that an error has occurred. This is done on a per error line basis, and allows other error indications that occur to be captured while the first error is being classified.

Detailed Description Text (21):
Each bit in the SCAPTURE register 72 has an associated bit in two of three mask registers, the associated bits indicating into which levels the particular error is to be classified. These registers are labelled L1 MASK 74, L2 MASK 76, and L23 MASK 78. The SCAPTURE register 72, in the preferred embodiment, is 36 bits wide, while the L1 MASK, L2 MASK and L23 MASK registers 74, 76, and 78 are 24, 24, and 12 bits wide, respectively.

Detailed Description Text (22):
The bit pattern stored in the L1 MASK register 74 comprises a conventional digital mask; that is, a "1" in a particular bit position of the mask classifies an error signal in this bit position in the SCAPTURE register 72 as a level 1 error, while a "0" indicates that the error is not to be classified in level 1. In the preferred embodiment, the L1 MASK register 74 applies to the most significant 24 bits of the SCAPTURE register 72. A mask in the L2 MASK register 76 also masks the most significant 24 bit positions in the SCAPTURE register 72 in the same manner as the L1 MASK in the L1 MASK register 74. This permits error indications captured in the SCAPTURE register 72 to initiate both level one and level two clock stopping, when necessary. This might be necessary when, for example, an error correction strategy calls for a serial processor and associated vector processor to stop synchronously and rapidly.

Detailed Description Text (23):
The L23 MASK of the L23 MASK register 78 operates on the least significant 12 bits of the SCAPTURE register 72 and classifies these bits as either level two or level three by the convention that a "1" indicates a level two error while a "0" indicates a level three error. The result is that only the least significant 12 bits of the SCAPTURE register 72 may be used for level three indications, while all 36 bits are permitted to classify associated errors as level 2.

Detailed Description Text (24):
The outputs of the SCAPTURE register 72 and the L1 MASK register 74 are connected to a level one detection circuit (LEVEL 1 DETECT) 80, which compares the registers contents on a bit-by-bit basis to determine whether any error signals in the SCAPTURE register 72 are to be classified as level one errors according to the L1 MASK in the L1 MASK register 74. If a bit in the most significant 24 bit positions of the SCAPTURE register 72 has been set by receipt of an error signal, and that bit is matched by a set bit in the corresponding bit position of the L1 MASK 74, the level one detection circuit 80 will output an L1 signal. Similarly, the outputs of the SCAPTURE register 72, the L2 MASK register 76, and the L23 MASK register 78 are all connected to a level two detection circuit 82 (LEVEL 2 DETECT). Any time an error indication in the SCAPTURE register 72 is masked by a corresponding L2 indication in a corresponding bit position in one of the registers 76 or 78, the circuit 82 outputs an L2 CHECK STOP signal.

Detailed Description Text (25):
As discussed above, SCAPTURE register bits which have their respective mask bits set in the L1 MASK register 74 will raise the unit's L1 signal, which is forwarded, on signal line 35-1 to the machine check collection circuit 52 in the CLKMAINT unit 50 (FIG. 3). As described below, these SCAPTURE bits also cooperate with corresponding inhibiting signals (INH) from an INHIBIT register 83 to inhibit any subsequent level two stop indications from being recognized by the level two detection circuit 82, since it would be possible to stop the unit out of synchronization with other units in the partition if an error indication was classified as a level two level in the cycle following the L1 classification.

Detailed Description Text (26):
Those bits of the SCAPTURE register 72 that are masked for level two are sent to the unit's gated clock generator to stop the unit, on the condition that the level two inhibiting function is not active. The possibility of asynchronous stopping between units is intrinsic with level two errors, so no blocking of clock stopping functions is done at this level. The detection of a subsequent level one check in the two cycles before the clocks are stopped in response to a level two error will initiate a partitioned clock stopping even though this level one error may have resulted from the level two error condition. This is done to prevent the error being dispersed through other units.

Detailed Description Text (27):
The SCAPTURE register 72 allows each error indication to be captured and held, but provides no information as to the sequence of error occurrence. The secondary lockout (SLOCKOUT) register 86, in combination with level lockout gating circuit 84 and a DETECT LOCKOUT circuit 88, performs this function by monitoring the outputs of the SCAPTURE register 72 for the presence of error conditions in much the same fashion as the SCAPTURE register 72 does with error signals. However, instead of locking each bit position as an error indication occurs as in the line lockout circuit, all positions within group levels are blocked in the SLOCKOUT register 86 after detection of the first error in any position. In this regard, the outputs of the SCAPTURE register 72 are fed through the level lockout gating circuit 84 to the SLOCKOUT register 86. In turn, the outputs of the SLOCKOUT register 86 are fed to the detect lockout circuit 88, which also receives the outputs of the L1 MASK register 74, the L2 MASK register 76, and the L23 MASK register 78. When the first clock stop error (classified as L1 or L2) is entered into the SCAPTURE register 72, the error is provided through the level lockout gating 84 to the input of the SLOCKOUT register 86. The error indication bit is compared in the DETECT LOCKOUT circuit 88 with all of the masks in the mask registers 74, 76 and 78. If the error indication occupies a bit position masked by a bit in the L1 or L2 masks, the DETECT

LOCKOUT circuit 88 provides a lockout signal to the level lockout gating 84, preventing it from providing any more error signals masked by bits in the L1 or L2 masks. Similarly, the first error indication entered in the SCAPTURE register 72 which is classified as an L3 error enters the SLOCKOUT register 86, and the detect lockout circuit 88 provides an L3 lock to prevent any more L3-classified error indications from entering the SLOCKOUT register 86. Thus, the SLOCKOUT register 86 will capture the first error indication classified as a L1 or L2 error, as well as the first non-clock check stop (L3 error). This permits error detection and recovery procedures to narrow down the investigation into the reason for an error occurrence. Finally, the card error communications collection function is performed by the check reporting circuit 90 which operates in a manner essentially equivalent to the lockout detection circuit 88 to provide an indication of a clock-stopping L1 or L2 error and an L3 error.

Detailed Description Text (28):
A circuit-specific implementation of the SECMAINT apparatus of FIG. 6 is illustrated in detail in FIG. 7 for one error indication bit. It is asserted that the logic of FIG. 7 is conventionally replicated in parallel to provide, for example, a 36-bit wide capability in the preferred embodiment. Once again, the circuit technology of FIG. 7 utilizes SRL-based logic. Thus, all registers are composed of arrays of parallel SRL's. In FIG. 7, the nth cells of the SCAPTURE, SCHKENBL, L1 MASK, and L2 MASK registers are indicated by 72-n, 73-n, 74-n, and 76-n, respectively, while the nth bit of the SLOCKOUT register is indicated by 86-n. Gates designated in FIG. 7 as AO are well known AND-OR gates in which two parallel, multi-input AND gates have their outputs connected as the only two inputs to an associated OR gate.

Detailed Description Text (29):
Assume now, that error indication bit n rises, signifying an error indication signal being provided by the error indicating unit connected to error input line n. Assume further that L1 MASK bit n is set, as is the nth bit of the SCHKENBL register. In this case, the output of the upper AND gate of AO gate 70 will rise, causing the output of the AO circuit 70 to rise. The level rise will be available on the positive output of cell 72-n of the SCAPTURE register 72. The positive output will be fed back to the AND gate of AO gate 70, preventing any successive error clocked into cell 72-n. In addition, the positive output of the SCAPTURE cell 72-n is fed to a gate combination including AND gate 100 and 24-way OR gate 101. The AND/OR combination 100 and 101 represents the level one detection circuit 80, with the OR gate 101 collecting the outputs of 23 other AND gates in addition to AND gate 100. If the L1 MASK bit corresponding to the nth bit of the SCAPTURE register is set, the AND gate 100 will be activated when the output of SCAPTURE cell 72-n rises. This essentially classifies the nth error indication as a level one error, which is forwarded by way of the OR gate 101 as an L1 signal. Similarly, if the bit in the L2 MASK corresponding to the nth cell of the SCAPTURE register is set, the output of the AND gate 110 will rise, causing the output of the 36-way NOR(N) gate 111 to fall, thereby providing the level two check stop signal in the proper polarity for interrupting the provision of clock signals. The NOR gate 111, the AND gate 110 and 35 other AND gates comprise the level two detect circuit 82 of FIG. 6.

Detailed Description Text (30):
The level lockout gate in circuit 84 is represented by the AO gate 120 whose output is connected to the input of the nth cell of the SLOCKOUT register 86. When an error signal is asserted on the nth error indication line, causing the output of SCAPTURE cell 72-n to rise, the output of the AO gate 120 rises. The rising output is clocked through SLOCKOUT cell 86-n. The output of this cell is fed back through the AO gate 120, which "locks" the cell from responding to any further changes in the output of SCAPTURE cell 72-n. The error indication locked into SLOCKOUT cell 86-n is classified as a level one/level two lockout through the combination of OR gate 122, AND gate 123, and 24-way OR gate 124. Similarly, the AND gate 126 and 12-way OR gate 127 classify the lockout as a level three lockout if the corresponding bit of the L23 MASK is a "0", which, when provided in inverted form to the AND gate 126, appears positive. Lockout detection of the bit in SLOCKOUT cell 86-n is provided by OR gate 130, AO gate 132 and inverter 134.

Detailed Description Text (32):
The INH register 83 of FIGS. 3 and 6 stores an inhibit mask equal in length to the

L1 mask, and operating on the same bits in the SCAPTURE register. Whenever one of the inhibit mask bits is set, the clock generator for the affected unit will not respond to an L2 CHECK STOP generated for a corresponding SCAPTURE register bit. For a detailed understanding, refer to FIGS. 4A and 7. Assume inhibit mask bit n is set, indicated by a positive level value for the output of INH register output 83-n (ENABLE L2 INHIBIT n). The output of the AND gate 102 will rise when SCAPTURE bit 72-n is set. This will deactivate the output of NOR gate 103 (INHIBIT L2 STOP), which, since the L2 CHECK STOP and INHIBIT L2 STOP signals are fed to the AND gate 104 in the clock generator (FIG. 4A) the INHIBIT L2 STOP signal will prevent the output (B) of the AO gate 68 from falling. This will limit the clock generator to respond only to the CARD RUN GATE signal for suspending provision of C.sub.1 and C.sub.2.

Detailed Description Text (34):
The CLKMAINT mechanism is shown in detail in FIGS. 8 and 9. In FIG. 8, the outputs of all of the level one detection circuits in the processor units are collected on input lines. Thus, for example, the signal lines 35-1 and 45-1 are shown. According to the L1CHKENBL MASK in the register 150, those L1 signals with a corresponding MASK bit pass through the gating circuit 152 to an L1CAPTURE register 154. The L1CAPTURE register 154 operates in the same manner as the SCAPTURE register described above. That is, it captures any L1 signal arriving on an L1 signal line from a unit, so long as that L1 signal line is enabled by the mask. A lockout loop consisting of a lockout gating circuit 155 and a L1 lockout register 153 captures the first L1 signal entering the L1 capture register 154 on any L1 signal line and then ignores any later arriving L1 signals on that line. The L1CAPTURE register is a t bit register, in which each bit position corresponds to a respective one of a maximum of t SECMAINT circuits.

Detailed Description Text (35):
The output of the L1CAPTURE register 154 is provided to each of four partition detection circuits 156-159. The partition detection circuits also receive at their inputs the output of a primary check stop circuit consisting of SRL pairs 160-0 through 160-t. Each partition detection circuit is for generating a signal identifying one of four predetermined partitions. Each SRL pair is preset with a two-bit code corresponding to one of the four predetermined partitions. Since each SRL provides both a positive and inverted output, each SRL pair of the primary check stop circuit indicates one state of a four-state code. Each primary check stop SRL pair is associated in each partitioned detection circuit with a corresponding bit in the L1 capture register 154. Thus, the output of bit position 0 of the L1 capture register 154 is associated in each of the partition detection circuits 156-159 with primary check stop SRL pair 160-0, and so on. Continuing the example, if a level one error signal is registered into bit position 0 of the LICAPTURE register 154 from the SECMAINT circuit connected to the input of bit position 0, it will be passed to the input of the partition detection circuit enabled by the code in SRL pair 160-0. The outputs of the partition detection circuits 156-159 provide board partition stop signals, PARTSTOP 0-3. Each PARTSTOP signal indicates that the clocks for each unit of a respective partition are to be interrupted. The partition detection circuit outputs are fed to a stop collection multiplexer 170, which is illustrated in FIG. 9. In FIG. 9, four board partition (BRDPART) registers 172-175 are connected to the inputs of the multiplexer 170. Each of the registers 172-175 has a partition mask in it in which each bit position corresponds to a respective unit. A mask defines a partition as a group of units whose clocks are to be interrupted in synchronism and in response to an L1 error signal from any member of the partition. The mask comprises a conventional 1-dimensional bit array with 1's entered into the bit position corresponding to partition members, and 0's into every other position. It should be obvious that the bit positions containing 0's are all of the remaining units of the processor which are not included in the partition.

Detailed Description Text (36):
The output of the stop collection multiplexer 170 is inverted and fed to a primary run gate (PRUNGATE) register 182 through a control circuit 180. When an L1 error occurs, a PARTSTOP signal is activated, causing the multiplexer 170 to select the output of the corresponding BRDPART register, the output being passed through the control circuit 180 and entered into the PRUNGATE register 182. The selected mask in the PRUNGATE register defines the CARD RUN GATE signals provided to the unit gated

clock generators. When a partition mask is selected and entered into the PRUNGATE register 182, the CARD RUN GATE signal for each unit in the partition will be deactivated by the 0 in the unit's bit position, thereby synchronizing the interruption of the units in the partition.

Detailed Description Text (37):
FIGS. 10 and 11 illustrate the SRL circuit implementation of the CLKMAINT apparatus of FIGS. 8 and 9. FIGS. 10 and 11 represent a single-bit slice of the layout illustrated in FIGS. 8 and 9, it being understood that the CLKMAINT apparatus comprises an integrated plurality of these slices operating cooperatively. In FIG. 10, a level one error signal is received from unit n and latched into L1CAPTURE cell 154-n. The input to the cell 154-n includes an AND/OR gate which is controlled by bit 150-n from the L1CHKENBL register 150. The integrated AO gate locks L1CAPTURE cell 154-n when the first L1 signal is latched into it. The output of L1CAPTURE cell 154-n is fed to the input of four partition detection circuits, each consisting of an AND (A) gate feeding an OR (O) gate. Each of the partition stop AND gates 200, 202, 204 and 206 receives also a respective combination of the outputs of check stop SRL pair 160-n. Thus, according to the state of the check stop pair SRL 160-n, the L1 error signal in L1CAPTURE cell 154-n will be fed forward from an AND gate to one of the OR gate 201, 203, 205 or 207 to form a partition stop signal.

Detailed Description Text (39):
In FIG. 11, the stop collection multiplexer consist of an array of AND gates 220-0 through 220-3, each receiving a respective PARTSTOP signal, as well as bit n from the board partition mask registers 172-175. An OR gate 221 collects the outputs of the AND gates 220-0 through 220-3, and forwards the partition mask bit selected by the AND gate array to a control circuit AND/OR-invert gate 223. The AOI gate 223 controls the input to SRL pair 182-n of the PRUNGATE register 182, and thus controls the state of the nth CARD RUN GATE signal. Also feeding the AOI gate 223 are a START CLKS signal through an inverting AND gate 225 and a STOP CLKS signal through an AND gate 227. The gates 225 and 227 receive the nth bit of the start-stop mask register 210 (FIG. 9). To initialize operation of the apparatus of this invention, the START CLKS signal is raised, causing the START-STOP MASK bit n to be entered into the nth bit of the PRUNGATE register 182. Once initially configured, the CARD RUN GATE signal provided by the PRUNGATE register 182 will continue to be asserted, until changed when an L1 error signal selects a partition mask, which is entered into the PRUNGATE register to determine the state of the CARD RUN GATE signals thereafter.

Detailed Description Text (40):
Inspection of FIGS. 7, 10, 11 and 4A will confirm that clock interruption in response to a L1 error is indeed completed in four machine cycles. An error indication is presented to SCAPTURE cell 72n during machine cycle 0 and is available from the output of the SRL at the beginning of the machine cycle 1. The error is classified as a level one error by gates 100 and 101, and is entered into L1 capture cell 154n. The L1 signal is available from L1 capture cell 154n at the end of cycle two. The captured level one error signal selects a partition group by propagation through one of the partition detection AND/OR combinations, and selects a partition mask through one of the stop collection AND gates. The selected partition mask is presented to the PRUNGATE register 182 and is latched during cycle 3. Finally, the CARD RUN GATE signal resulting from latching of the partition mask into the PRUNGATE register 182 causes a change in the CLKRUN signal output from the SRL 69 at the end of cycle 4. Thus, with detection of the level one error in cycle 0, all units in the partition selected by the level one signal are interrupted simultaneously at the end of cycle 4.

Detailed Description Text (41):
Referring now to FIGS. 3, 6, 8 and 9, assume the occurrence of a level 2 error in the unit denoted by reference numeral 30. In this regard, one of the modules 32 raises an error indication on one of the error lines 71-1 through 71-q which is gated into the SCAPTURE register 72 through the lockout gate 71. Assuming the error is the first occurring since initialization, it will be entered into the lockout register 86 at a bit position dedicated to the module which originated it. At the same time, the error will be output by the SCAPTURE register at a corresponding bit position and classified by the detection circuits in response to the L1, L2, and L23 masks. Once classified, the L2 CHECK STOP signal is raised, resulting in suspension

of the unit clocks. The check reporting circuit 90 raises a level 1/level 2 lockout signal at the output of the OR gate 127 (FIG. 7), which is forwarded to the primary communications collector 58. The primary communications collector 58 operates conventionally to generate an INTERRUPT signal forwarded to the support processor 59. In response to the INTERRUPT signal, the support processor scans the SECMAINT and CLKCNTL lockout in capture registers to initiate appropriate diagnostic and corrective action.

Detailed Description Text (42):
Assuming that the error indication in the just-related example is classified as a level 1 error, the L1 signal would be output by the SECMAINT unit on signal line 35-1, on which it will be transferred to the CLKMAINT unit. There, the L1 signal is gated through the lockout gating 152 into a bit position in the LICAPTURE register 154 (and a corresponding bit position in the LILOCKOUT register 153) as originating from the unit 30. In the CLKMAINT unit, the L1 signal is compared against the four-state code in the primary check stop SRL pair 160-0 to determine which one of four partitions the unit 30 is a part of. The appropriate PARTSTOP signal is provided from one of the four partition detection circuits 157-159, and selects one of the four board partition masks in the registers 172-175. The mask is entered into the PRUNGATE register 182, with the result that the card run gate signals for all units in the partition are deactivated, thereby simultaneously and synchronously suspending the operations of the selected partition.

Detailed Description Text (43):
Continuing the example of L1 partitioned clock stopping, the clock signals in the unit where the L1 error originated might already have been stopped if the error is also classified as an L2 error, and if the mask in the units inhibit register does not prevent the suspension of clocks in response to the L2 check stop signal. In this case, the clocks in the unit whereat the error indication originated have been interrupted two machine cycles before the other units in the affected partition.

Detailed Description Text (45):
Rapid maintenance access into the malfunctioning processor can be provided to the support processor 59 by an address map which embraces the check enable registers, mask registers, and primary check stop SRL latch pairs. Since these elements of the invention can be included in the addressable resource array of the support processor, they can be programmed directly by the support processor. Advantageously, this would enable the support processor to program the L1MASK, L2MASK, L23MASK, and the check enable mask in the SCHKENBL register 73 (FIG. 6). Additionally, a direct connection between the support processor and the inhibit register of the SECMAINT unit (FIG. 6) would allow the L2 inhibition pattern to be programmed by the support processor.

Detailed Description Text (46):
In the CLKMAINT unit (FIGS. 8 and 9), connection to the support processor 59 gives the ability to establish unit partitions by programming the partition mask registers 172-175. Further, each processor unit is assigned to a partition through the programming of the primary check stop SRL pairs. Thus, for example, the processor unit serviced by the SECMAINT unit 35 is linked to a partition by the four-state code programmed into SRL pair 160-0.

Detailed Description Text (47):
The support processor also is used to set the initial CLOCK RUN gate pattern in the PRUNGATE register 182, provides the start and stop signals to the control block 180 (FIG. 9) and programs the start-stop mask into the start-stop register of the CLKCNTL unit.

End of Result Set

L12: Entry 1 of 1 File: USPT Mar 10, 1998

DOCUMENT-IDENTIFIER: US 5727209 A
TITLE: Apparatus and method for achieving reduced overhead mutual-exclusion and maintaining coherency in a multiprocessor system utilizing execution history and thread monitoring

US PATENT NO. (1):
5727209

Detailed Description Text (35):
A dense per-thread bitmap data structure has an array of bits with one bit per thread. When a thread passes through a quiescent state since the data structure was reset, the corresponding bit is cleared.

Detailed Description Text (36):
A distributed per thread bitmap data structure embeds thread bits into a structure that facilitates thread creation and destruction. For example, a flag is used to track each thread in the data structure.

Detailed Description Text (37):
A hierarchical per-thread bitmap is a data structure that maintains a hierarchy of bits. The lowest level maintains one bit per thread. The next level up maintains one bit per group of threads and so on. All bits are preset to a predetermined state, for example, a one-state. When a thread is sensed in a quiescent state, its associated bit is set to a zero-state in the lowest level of the hierarchy. If all bits corresponding to threads in the same group are in a zero-state, the associated group bit in the next higher level is set to a zero-state and so on until either the top of the hierarchy or a non zero bit is encountered. This data structure efficiently tracks large numbers of threads. Massively parallel shared-memory multiprocessors should use a bitmap hierarchy mirroring their bus hierarchy.

Detailed Description Text (39):
Each bit is explicitly set to a predetermined state, preferably a one-state, by a reset signal 114.

Detailed Description Text (40):
Each bit (or group of bits for hierarchial bitmaps) has an associated generation counter. A global generation counter is incremented to reset summary of thread activity 106. When a thread is sensed in a quiescent state, and its associated bit is currently in a zero-state, its associated generation counter is compared with the global generation counter. If the counters differ, all bits associated with the quiescent thread are set to a one-state and the associated generation counter is set to equal the global counter.

Detailed Description Text (42):
A thread counter (not shown) may be used to indicate the number of threads remaining to be sensed in a quiescent state since the last reset signal 114. The thread counter is preset to the number of threads (or for hierarchial schemes, the number of threads in a group) and is decremented each time a thread bit is cleared. When the counter reaches zero, all threads have been sensed in a quiescent state since the last reset. If threads can be created and destroyed, the counters corresponding to the threads being destroyed must be decremented.

Detailed Description Text (43):
Callback processor 104 interfaces with the quiescence-indicating scheme chosen for summary of thread activity 106 and therefore has various possible implementations. For example, if the quiescence-indicating bits in summary of thread activity 106 have other purposes, no additional overhead need be incurred by callback processor 104 in checking them. Consider a data structure having a dedicated summary of thread activity and a per-thread bit for indicating the occurrence of some unusual condition. Any thread accessing the data structure must execute special-case steps in response to the per-thread bit such as recording its quiescence before accessing the data structure.

Detailed Description Text (46):
readers just before or just after accessing the data structure protected by mutual-exclusion mechanism 90 (invoking callback processor 104 just after accessing the data structure will incur overhead unless the quiescence-indicating bits in summary of thread activity 106 have multiple purposes);

Detailed Description Text (55):
A callback processor 124 includes a one-bit-per processor bitmask. Each bit indicates whether its associated processor 16 must be sensed in a quiescent state before the current generation can end. Each bit corresponds to a currently functioning processor and is set at the beginning of each generation. When each processor 16 senses the beginning of a new generation, its associated per-processor context switch counter 122 value is saved. As soon as the current value differs from the saved value, the associated bitmask bit is cleared indicating that the associated processor 16 is ready for the next generation.

Detailed Description Text (56):
Callback processor 124 is preferably invoked by a periodic scheduling interrupt 126. Processor 16 may alternatively clear its bitmask bit if scheduling interrupt 126 is in response to an idle loop, a user-level process execution, or a processor 16 being placed off line. The latter case is necessary to prevent an off line processor from stalling the callback mechanism and causing a deadlock.

Detailed Description Text (57):
When all bits in the bitmask are cleared, callback processor 124 processes all callbacks 128 in a global current generation 130 and all callbacks 128 associated with the current processor in a per-processor current generation 131.

Detailed Description Text (85):
rcc.sub.-- olmsk is a bitmask in which each bit indicates whether a corresponding processor 16 is on line;

Detailed Description Text (86):
rcc.sub.-- needctxtmask is a field implementing summary of execution history 138 (the field is a bitmask in which each bit indicates whether a corresponding processor 16 has been sensed in a quiescent state);

Detailed Description Text (114):
Callback processor 124 is implemented by a function referred to as rc.sub.-- chk.sub.-- callbacks. Callback processor 124 is invoked by interrupt 126, which is preferably a hardware scheduling clock interrupt referred to as hardclock(), but only if one or more of the following conditions are met: rclocknxtlist is not empty and rclockcurlist is empty (indicating that the current processor is tracking a generation of callbacks and there are callbacks ready to join a next generation); rclockcurlist is not empty and the corresponding generation has completed; or the bit in rcc.sub.-- needctxtmask that is associated with the current processor is set. The latter condition ensures that if there is a current generation of callbacks, the current processor must be in, or have passed through, a quiescent state before the generation can end.

Detailed Description Text (119):
if the bit in rcc.sub.-- needctxtmask corresponding to the current processor is not set, return (do not execute the following steps);

Detailed Description Text (125):
if the bit in rcc.sub.-- needctxtmask associated with the current processor is already cleared, release rcc.sub.-- mutex and return;

Detailed Description Text (126):
clear the bit in rcc.sub.-- needctxtmask associated with the current processor to indicate that the current processor has completed the current generation;

Detailed Description Text (128):
if any bit in rcc.sub.-- needctxtmask is still set, release rcc.sub.-- mutex and return;

Detailed Description Text (159):
if rcc.sub.-- needctxtmask has a bit set (current generation not complete) or if rcc.sub.-- maxgen is less than rcc.sub.-- curgen (specified generation complete), return; and

Detailed Description Text (180):
Handle table 170 accommodates NENTRIES 3 lock handles. A computer, such as computer 10, typically has a 32-bit (2 32) wide address bus which is sufficient to address 1024 NENTRIES (1024 3=2 30). Therefore, handle table 170 is adequately sized for all practical applications.

Detailed Description Text (187):
To deallocate data structure 210, it is first disassociated with its corresponding lock handle by linking the lock handle back on handle-free list 200. A reader attempting to use data structure 210 during lock handle disassociation will encounter either the next entry in handle-free list 200 or a pointer to data structure 210. Free list 200 entries are distinguished from internal data structure pointers by examining a bit alignment of the returned value. In this implementation, internal data structure pointers are aligned on a 4-byte boundary by setting their least significant two bits to zero, whereas handle-free list entries have their least significant pointer bit set to one (the associated lock handle index is shifted left one position to compensate). Therefore, a reader encountering a value with the least significant bit set knows that the associated lock handle does not correspond to internal data structure 210.

Detailed Description Text (188):
A reader encountering data structure 210 locks data structure 210 and checks a flag word embedded in data structure 210. When data structure 210 is prepared for deallocation, a "DEAD" flag bit is set therein, and data structure 210 is placed on a "pending deallocation" list of data structures. The DEAD flag bit is set under protection of a per-data structure lock.

Detailed Description Text (189):
The reader encounter data structure 210 with the DEAD flag bit set informs its controlling process that data structure 210 is pending deallocation. In this implementation, a lookup routine uses the existing per-data structure lock upon sensing the DEAD bit set, releases the lock, and informs its controlling process that the lock handle is no longer associated with an active internal data structure.